

1 What is claimed is:

2 1. A method for mapping procedural code to object-oriented classes,
3 comprising:

4 starting a graphical user interface (“GUI”) in a procedural
5 programming language process space, wherein a user enters a command
6 through the GUI;

7 initializing a mapping layer in an object-oriented programming
8 language process space, wherein the mapping layer comprises entry-points
9 that have corresponding algorithms that invoke object-oriented class
10 instantiation methods and/or remote method invocations (“RMIs”);

14 invoking one of the entry-points; and

15 the mapping layer executing an algorithm corresponding to the
16 invoked entry-point.

18 2. The method of claim 1, wherein executing the algorithm comprises invoking
19 a class instantiation method.

21 3. The method of claim 1, wherein executing the algorithm comprises invoking
22 an RMI.

24 4. The method of claim 1, wherein the procedural programming language is
25 C++.

27 5. The method of claim 1, wherein the object-oriented programming language
28 is Java.

30 6. The method of claim 5, wherein the mapping layer is accessed through a
31 Java Native Interface (“JNI”) and invoking one of the entry-points comprises:

32 invoking a JNI application programming interface (“API”) call,
33 wherein the JNI API call invokes one of the entry-points

1 7. The method of claim 6, wherein the mapping layer is proxied in the
2 procedural programming language process space by a proxy object that includes
3 proxy object methods corresponding to the entry-points, and executing a GUI
4 callback further comprises:

5 invoking one of the proxy object methods, wherein the invoked
6 proxy object method performs the invoking one of the entry-points step.

7

8 8. The method of claim 1, wherein the entry-points are methods of the mapping
9 layer.

10

11 9. The method of claim 1, further comprising returning data to the procedural
12 programming language process space.

13

14 10. A computer readable medium containing instructions for mapping
15 procedural code to object-oriented classes, by:

16 starting a graphical user interface (“GUI”) in a procedural
17 programming language process space, wherein a user enters a command
18 through the GUI;

19 initializing a mapping layer in an object-oriented programming
20 language process space, wherein the mapping layer comprises entry-points
21 that have corresponding algorithms that invoke object-oriented class
22 instantiation methods and/or remote method invocations (“RMIs”);

23 executing a GUI callback in response to the command, wherein the
24 GUI callback comprises procedural code and wherein executing a GUI
25 callback in response to the command comprises:

26 invoking one of the entry-points; and

27 the mapping layer executing an algorithm corresponding to the
28 invoked entry-point.

29

30 11. The computer readable medium of claim 10, wherein executing the
31 algorithm comprises invoking a class instantiation method.

32

33 12. The computer readable medium of claim 10, wherein executing the
34 algorithm comprises invoking an RMI.

1
2 13. The computer readable medium of claim 10, wherein the procedural
3 programming language is C++.

4
5 14. The computer readable medium of claim 10, wherein the object-oriented
6 programming language is Java.

7
8 15. The computer readable medium of claim 14, wherein the mapping layer is
9 accessed through a Java Native Interface (“JNI”) and invoking one of the entry-
10 points comprises:

11 invoking a JNI application programming interface (“API”) call,
12 wherein the JNI API call invokes one of the entry-points.

13
14 16. A computer system that enables the mapping of procedural code to object-
15 oriented classes, comprising:

16 a memory;
17 a processor that runs an application, wherein the application
18 generates:

19 a graphical user interface (“GUI”) in a procedural
20 programming language process space, wherein users enter
21 commands through the GUI; and,

22 a mapping layer in an object-oriented programming language
23 process space, wherein the mapping layer comprises entry-points that
24 have corresponding algorithms that invoke object-oriented class
25 instantiation methods and/or remote method invocations (“RMIs”).

26
27 17. The computer system of claim 16, wherein the GUI executes callback code
28 in response to an entered command and the executed callback code invokes one of
29 the mapping layer entry-points.

30
31 18. The computer system of claim 16, wherein the entry-points are mapping
32 layer methods that are accessed from the procedural programming language process
33 space through application programming interface (“API”) calls.

1 19. The computer system of claim 16, wherein the procedural programming
2 language is C++.

3

4 20. The computer system of claim 16, wherein the object-oriented programming
5 language is Java and the object-oriented programming language process space is a
6 Java Virtual Machine.